



# PRODUCT UPDATE

ATARI HOME COMPUTER SYSTEM

**Page 1.** This definition is missing from the **TERMINOLOGY** list:

**Page 6.** This information pertains to the **ARITHMETIC OPERATORS** subtraction and exponentiation:

**Page 7.** This Note regards the use of the **LOGICAL OPERATORS**:

**Page 13.** This Note is in reference to **SCREEN EDITING**:

**Page 20.** This Note regards **ON/GOSUB** statements:

**Page 22.** Further information on **RESTORE (RES.)**:

**Page 25.** Some additional information on using the **INPUT (I.)** statement:

## ATARI® BASIC Reference Manual Update

This product update contains a number of corrections and additions to the *ATARI BASIC Reference Manual*.

**Floating Point Number:** A number containing an integer part, a decimal point, and a fractional part. The total number of significant digits in a floating point number, excluding the exponent, may be either nine or ten. This depends on whether the exponent is an even or odd multiple of 10.

**Note:** Avoid negating zero, as this will produce an invalid number. For example, if you type

```
PRINT -0
the result will be
-0E- <8
```

**Note:** Since the algorithm used to generate exponents (^) is only an approximation, you cannot obtain integer results with it—for example,  $2^2 = 3.99999996$ . To correct this, use the following technique:

```
X = 2 ^ 2
PRINT INT (X + .5)
4
```

**Note:** Avoid using the statement `PRINT A=NOT B`, as the results are not predictable. Essentially, any `PRINT` statement with a `NOT` operator will be unpredictable.

**Note:** Large amounts of editing may lock up the system. It's recommended that programs under development be stored to cassette or diskette periodically (every 30 or 40 edits) with the `SAVE` or `CSAVE` command.

**Note:** If an `ON/GOSUB` expression evaluates to a number greater than the number of subroutine entries, then a `POP` statement will be necessary to clear the stack (see `POP` command, Section 4).

The `RESTORE` statement will not generate an error if the line number referenced does not exist. Instead it will `RESTORE` to the next larger line number in the program. Care should be taken to update `RESTORE` statements when renumbering a BASIC program.

When executing an `INPUT` from the screen, avoid moving the cursor away from and then back to the same line; otherwise, the wrong data may be input. Specifically, the `INPUT` prompt will be included in the `INPUT` string.

If a string of 128-255 characters is `INPUT`, then RAM locations 1536-1664 will be overwritten. This area is normally reserved for storage of programs or data. (See the *ATARI Tech Reference Notes*.)

To `INPUT` strings of more than 127 characters, use the `GET` command and store the values into a string (see Section 5, `OPEN/CLOSE` and `PUT/GET` commands).

**Note:** The maximum number of characters that can be `INPUT` from the screen is 120. The maximum for other devices is 255.

**Note:** Make sure that every `INPUT` statement has a variable after it; otherwise, unpredictable results may occur.



**Page 26.** This regards the use of the **LOAD (LO.)** command:

This Note should follow the **LPRINT (LP.)** command description:

**Page 27.** This information pertains to the file spec definition:

**Page 28.** This is an addition to the **POINT (P.)** section:

In the last paragraph under **PRINT (PR. or ?)**, the first sentence should read:

The following sentence should conclude the final paragraph on **PRINT (PR. or ?)**:

This note should then conclude this section on **PRINT (PR. or ?)**:

This Note regards the **PUT (PU.)/GET (GE.)** section:

**Page 30.** Here is a corrected version of the table—note in particular the correction on cmdno 32:

**Note:** If a program is loaded that is too large for the available memory space, it may give unpredictable results without an error message.

**Note:** An LPRINT command with a semicolon at the end will cause the following LPRINT statement to print on the next 40-column tab. A 40-column printer will move to the next line in such a case. To use the semicolon effectively, use the OPEN statement for the printer, then write to the printer with a PRINT statement (see OPEN/CLOSE and PRINT commands, Section 5).

**Note:** Be sure to include the closing quotation marks on a filespec parameter, especially when putting multiple statements on one line. For example,

```
OPEN #1, 4, 0, "D:TEST":STOP
will work, but
OPEN #1, 4, 0, "D:TEST:STOP
will not function correctly.
```

**Note:** To update a file, you must open it with a 12 in aexp1.

A comma tabs every 10 spaces.

However, if the last character to be printed (as in a string with quotation marks) is a **CTRL R** or **CTRL U**, then the next PRINT will begin at the end of the current line.

**Note:** In rare circumstances data printed to a diskette may have part of the BASIC program embedded in it. If this occurs, retry the operation.

**Note:** In certain circumstances the GET function may modify other variables within the program. To avoid this, PRINT any number to the screen between each GET.

cmdno	OPERATION	EXAMPLE
3	OPEN	Same as BASIC OPEN
12	CLOSE	Same as BASIC CLOSE
13	STATUS REQUEST	Same as BASIC STATUS
17	DRAW LINE	Same as BASIC DRAWTO
18	FILL	See Section 9
32	RENAME	XIO 32,#1,0,0,"D:TEMP,CAROL"
33	DELETE	XIO 33,#1,0,0,"D:TEMP.BAS"
35	LOCK FILE	XIO 35,#1,0,0,"D:TEMP.BAS"
36	UNLOCK FILE	XIO 36,#1,0,0,"D:TEMP.BAS"
37	POINT	XIO 37,#1,A,B
38	NOTE	XIO 38,#1,X,Y
254	FORMAT	XIO 254,#1,0,0,"D2:"



**Page 33.** The last sentence in the paragraph about the **CLOG** function should read:

**Page 34.** The last sentence in the paragraph about the **LOG** function should read:

**Page 38.** The last line in the first paragraph should read:

**Page 39.** The first sentence should read:

In the second paragraph, the last line should read:

This is additional information on the **VAL** function:

This information pertains to **String Concatenation**:

In *Figure 7-6*, the correct result of the program on the left is:

**Page 42.** Some additional information on using the **DIM (DI.)** statement:

**Page 43.** This is an additional Note for the **DIM (DI.)** section:

Additional information on using the **CLR** command:

CLOG(0) through CLOG(1) are inaccurate and should not be used.

LOG(0) through LOG(1) are inaccurate and should not be used.

was stored there previously.

Upon execution, the screen displays THE SQUARE ROOT OF 10000 IS 100.

number 1000000000.

Only the numeric field will be translated, while the text will be ignored. For example:

VAL("5SUM")=5

**Note:** BASIC cannot move strings of 256-character multiples correctly. String lengths should be checked; if any string contains a multiple of 256 characters, add or subtract one character from the amount to be moved.

BCD#

Make sure that the DIM statement does not contain a space between the string or array name and the left parenthesis of the dimensioned amount; otherwise, the following will happen—

DIM L (10) becomes DIM L10)

—and this variable can no longer be referenced.

**Note:** The command COM is identical to DIM and may be used in its place.

**Note:** Due to a discrepancy in boundary checking, arrays of up to 32766 by 32766 in size can be dimensioned. The programmer should size the array ahead of time to ensure that there is no "virtual" storage space.

The second sentence in the last paragraph, beginning "It also clears ...," should be deleted.

The CLR command will not initialize the values in strings and arrays.



Page 45. Here is a corrected version of TABLE 9.1:

TABLE 9.1—TABLE OF MODES AND SCREEN FORMATS

Gr. Mode	Mode Type	Horiz. (Columns)	SCREEN FORMAT		Number Of Color Registers	Split Screen	RAM Required (Bytes) Full Screen
			Vert. (Rows) Split Screen	Vert. (Rows) Full Screen			
0	TEXT	40	—	24	1½	—	992
1	TEXT	20	20	24	5	674	672
2	TEXT	20	10	12	5	424	420
3	GRAPHICS	40	20	24	4	434	432
4	GRAPHICS	80	40	48	2	694	696
5	GRAPHICS	80	40	48	4	1174	1176
6	GRAPHICS	160	80	96	2	2174	2184
7	GRAPHICS	160	80	96	4	4190	4200
8	GRAPHICS	320	160	192	1½	8112	8138

Page 49. The last sentence under PLOT (PL.) should read:

“The range of points begins at 0 and extends....”

Page 50.

In TABLE 9.3, the color PURPLE should be inserted after PINK in the first column, and the number 5 should be inserted after 4 in the second column.

Page 51. The sentence directly under TABLE 9.4 should read:

“DEFAULT” occurs if no SETCOLOR statement is used.

Page 53. Here is a corrected version of TABLE 9.5:

MODE, SETCOLOR, COLOR TABLE

Default Colors	Mode or Condition	SETCOLOR (aexp1) Color Register No.	Color (aexp)	DESCRIPTION AND COMMENTS
LIGHT BLUE	Mode 0 and all text windows	0	Color data actually determines character to be printed.	—
DARK BLUE		1		Character luminance (same color as background)
BLACK		2		Background
		3		—
		4		Border
ORANGE	Modes 1 and 2 (text modes)	0	Color data actually determines character to be printed.	Character
LIGHT GREEN		1		Character
DARK BLUE		2		Character
RED		3		Character
BLACK		4		Background, border
ORANGE	Modes 3, 5, and 7 (four-color modes)	0	1	Graphics point
LIGHT GREEN		1	2	Graphics point
DARK BLUE		2	3	Graphics point
		—	—	—
BLACK		4	0	Graphics point (background default), border
ORANGE	Modes 4 and 6 (two-color modes)	0	1	Graphics point
		—	—	—
		—	—	—
BLACK		4	0	Graphics point (background default), border
LIGHT BLUE	Mode 8 (1 color, 2 luminances)	—	—	—
		1	1	Graphics point luminance (same color as background)
DARK BLUE		2	0	Graphics point (background default)
BLACK		—	—	—
		4	—	Border



**Page 54.** In *Figure 9-4*, line 80 should read:

**Page 55.** This information pertains to **TABLE 9.6:**

**Page 56.** Here is a corrected version of **TABLE 9.7:**

**Page 58.** The last paragraph should read as follows:

In **TABLE 10.1:**

**Page 63.** The last line in item 9 should read:

**Page 67.** In *Figure 11-2*, line 0260 under Data should be:

**Page E-1.**

**Page H-7.** Line 160 in the program should read:

**Page H-8.** Line 50 in the program should read:

**Page 117.**

**Page 118.**

**Page 119.**

80 XIO 18, #6, 12, 0, "S:"

In Column 1, # 14, a period, not a bar, shows on the screen.

In Column 3, #'s 92-95 should show a superscripted circled 1 next to their characters.

**TABLE 9.7—CHARACTER/COLOR ASSIGNMENT**

		Column 1 Conversion	Column 2 Conversion	Column 3 Conversion	Column 4 Conversion
<b>MODE 0</b>	<sup>2</sup> SETCOLOR 2	# + 32	# + 32	#-64	NONE
		POKE 756,224		POKE 756,226	
<b>MODE 1</b>	SETCOLOR 0	# + 32	# + 32	#-32	#-32
<b>OR</b>	SETCOLOR 1	NONE	# + 64	#-64	NONE
<b>MODE 2</b>	SETCOLOR 2	# + 160	# + 160	# + 96	#-96
	SETCOLOR 3	# + 128	# + 192	# + 64	# + 128

<sup>2</sup> Luminance controlled by SETCOLOR 1, 0, LUM.

Note that the DATA statement in line 90 ends with 256, which is outside of the designated range. The 256 is.....

The PITCH VALUE of 193 should have a musical note of "E," not "D."

precedence will save a few bytes.

#2

The right parentheses are missing after the word "CONSTANT" in Atari Functions of Inverse Cosine, Inverse Secant, and Inverse Cosecant.

160 IF K= 125 OR K= 155 THEN 180

50 PLOT 0,0:DRAWTO 159, DR

Following COM, "{see DIM}" should be deleted and replaced with "A-1."

Under "Input/Output Devices," Line Printer should be followed by "(P:)," not "(L:)."

"NOTE, 26" is missing from the listing.



Page 120.

INSIDE BACK COVER  
here is the corrected  
table:

"STATUS, 29" is missing from the sublisting under "Statement" and also from the regular listing.

#### MODE, SETCOLOR, COLOR TABLE

Default Colors	Mode or Condition	SETCOLOR (aexp1) Color Register No.	Color (aexp)	DESCRIPTION AND COMMENTS
LIGHT BLUE	Mode 0 and all text windows	0	Color data actually determines character to be printed.	—
DARK BLUE		1		Character luminance (same color as background)
		2		Background
BLACK		3		—
		4		Border
ORANGE	Modes 1 and 2 (text modes)	0	Color data actually determines character to be printed.	Character
LIGHT GREEN		1		Character
DARK BLUE		2		Character
RED		3		Character
BLACK		4		Background, border
ORANGE	Modes 3, 5, and 7 (four-color modes)	0	1	Graphics point
LIGHT GREEN		1	2	Graphics point
DARK BLUE		2	3	Graphics point
		—	—	—
BLACK		4	0	Graphics point (background default), border
ORANGE	Modes 4 and 6 (two-color modes)	0	1	Graphics point
		—	—	—
		—	—	—
		—	—	—
BLACK		4	0	Graphics point (background default), border
LIGHT BLUE	Mode 8 (1 color, 2 luminances)	—	—	—
		1	1	Graphics point luminance (same color as background)
DARK BLUE		2	0	Graphics point (background default)
		—	—	—
BLACK		4	—	Border